Guarding Against Universal Adversarial Perturbations in Data-driven Cloud/Edge Services

Xingyu Zhou, Robert Canady, Yi Li, Shunxing Bao, Yogesh Barve, Daniel Balasubramanian, Aniruddha Gokhale
Dept of Computer Science, Vanderbilt University, Nashville, TN 37235
Email: {xingyu.zhou,robert.e.canady,yi.li,shunxing.bao,yogesh.d.barve,daniel.a.balasubramanian,a.gokhale}@vanderbilt.edu

***Abstract*—Although machine learning (ML)-based models are increasingly being used by cloud-based data-driven services, two key problems exist when used at the edge. First, the size and complexity of these models hampers their deployment at the edge, where heterogeneity of resource types and constraints on resources is the norm. Second, ML models are known to be vulnerable to adversarial perturbations. To address the edge deployment issue, model compression techniques, especially model quantization, have shown significant promise. However, the adversarial robustness of such quantized models remains mostly an open problem. To address this challenge, this paper investigates whether quantized models with different precision levels can be vulnerable to the same universal adversarial perturbation (UAP). Based on these insights, the paper then presents a cloud-native service that generates and distributes adversarially robust compressed models deployable at the edge using a novel, defensive post-training quantization approach. Experimental evaluations reveal that although quantized models are vulnerable to UAPs, post-training quantization on the synthesized, adversarially-trained models are effective against such UAPs. Furthermore, deployments on heterogeneous edge devices with flexible quantization settings are efficient thereby paving the way in realizing adversarially robust data-driven cloud/edge services.**

***Index Terms*—Adversarial machine learning, universal perturbation, edge computing, quantization, hardware acceleration.**

I. Introduction

Deep learning-based machine learning (ML) models are becoming commonplace in cloud-based services, such as in object detection, image classification, speech recognition, etc. However, due both to real-time response requirements and privacy concerns, many of these services, particularly those used in video surveillance or proactive maintenance in industrial plants, must rely on edge computing [1]. Edge services are designed and deployed in such a way that the critical, time-sensitive components execute at the edge where ML models are used for prediction tasks [2] and the remainder of the service components are deployed in the cloud.

Edge computing is, however, characterized by extreme heterogeneity in its resource types and significant constraints on their capacities (both compute and power). Consequently, the complex and large ML models that are traditionally utilized in cloud-based services cannot naïvely be deployed at the edge. Hence, ML model compression techniques, especially model quantization [3], are attracting significant attention. These methods support high compression ratios with limited performance loss, thereby enabling a practical cloud-edge application where full-precision models are deployed on cloud servers and lightweight, quantized models on edge devices.

Yet, the resource heterogeneity at the edge precludes a one-size-fits-all quantized ML model for a given full precision ML model; rather the quantized model's network architecture will vary based on the quantization settings [4]. To automate the transformation from full-scale to quantized models that are customized for the underlying edge hardware platform, solutions such as hardware-aware training [5] and model compilation [6] have been proposed. One recent work designs a model compression pipeline to generate a neural network and then specializes it for deployments across diverse platforms [7].

Despite the widespread use of ML models and their quantized counterparts at the edge [8], traditional ML models tend to be vulnerable to adversarial attacks [9], which are carefully designed inputs with bound-limited (i.e., small and almost undetectable) injected perturbations that can greatly mislead the pre-trained models into making suboptimal or incorrect inferences. Past research has even shown the existence of universal adversarial perturbations (UAP), where the same perturbation has the ability to adversarially impact almost all data samples for a certain task [10]. Moreover, these perturbations are also easily transferable to other models in an entirely black-box manner.

Although a significant amount of research on adversarial robustness (i.e., being able to defend against adversarial attacks) of traditional ML models exists, scarce amount of work exists on evaluating the robustness of quantized ML models [11]. The available literature indicates that model compression techniques set a limit on the amplitude of the classification prediction score[1] caused by the perturbation and therefore improve robustness, while other works even propose compression-based defense solutions [12].

Consequently, for applications that utilize the cloud/edge ML model deployment pattern where adaptive model compression is used for different types of devices at the edge as shown in Figure 1, we pose the following question for these range of deployments: *Are quantized models with different precision levels vulnerable to the same universal adversarial perturbations (UAP) as non-quantized models?* Answering this is vital because the existence of the same adversarial perturbations that can compromise models at all precision levels would

[1] A neural network makes a classification inference by choosing a class that has the highest amplitude of the prediction score. This is often realized by a Softmax function as the model output.

greatly reduce the threshold of potential adversarial threats in practice, thereby alleviating the need for designing point solutions for robustness and rather offer a general-purpose adversarial robustness service, which is the aim of this paper.

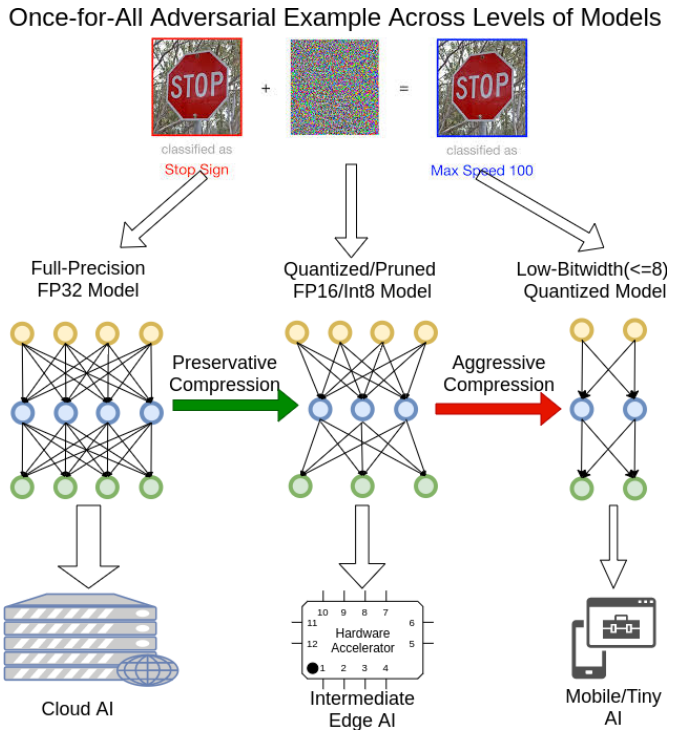

Figure 1: Practical Deployment of Hardware-aware Quantized Models under the Threat of Universal Adversarial Perturbations. The same perturbation may lead to model performance deviations on devices across multiple deployment levels.

To that end, we propose a cloud-native service comprising adaptive defensive ML model processing and deployment capabilities for cloud/edge applications under potential universal adversarial perturbation threats [10]. We propose an attack evaluation workflow to reveal the adversarial impact of UAPs across multiple representation precision levels. To solve this problem, we conduct a theoretical analysis on the impact of quantization on adversarial robustness and then develop a defensive post-training quantization of adversarially trained models, which then are deployed in the run-time infrastructure to realize adversarially robust cloud/edge ML-based services.

We evaluate our ideas for different precision levels of quantized neural networks using the CIFAR10/CIFAR100/SVHN object classification datasets, which are suitable for applications, such as surveillance or wildlife monitoring, and are the most widely used computer vision datasets. The experimental results using our evaluation workflow reveal two interesting observations. First, the natural quantized models (i.e., without adversarial robustness) remain vulnerable to universal perturbations and that the perturbations from some quantized models are even transferable to other quantized models. Second, the state-of-art adversarial training methods are efficient against universal perturbations. Moreover, adversarially-trained quantized models via post-training quantization preserves the robustness against these kinds of attacks and hence can be realized within a reusable and configurable, adversarial defense cloud-native service for edge-based applications. Doing so obviates the need for *reinventing the wheel* in realizing adversarially robust, data-driven edge services.

In summary, this paper makes the following contributions:

- We reveal the potential threat of universal adversarial perturbation (UAP) in cloud/edge applications and present an evaluation workflow to formalize the resilience testing of quantized neural network models under UAP attacks.
- We show the consistency of robustness of adversarially trained models and propose a defensive quantization method towards more robust quantized models that can be codified into a configurable and reusable solution that edge services can utilize.
- We evaluate the UAP attack and defense on the most widely used CIFAR10/CIFAR100/SVHN datasets. We show the vulnerability of normal quantized models and highlight the compactness and efficiency of our proposed defense strategies.

The rest of the paper is organized as follows. Section II provides background and related research comparing it to our work; Section III presents our proposed evaluation workflow and defense strategies; Section IV validates our approach using various settings of the CIFAR10/CIFAR100/SVHN datasets; Finally, we conclude in Section V and discuss future directions.

II. Background and Related Work

To make this paper self-contained, this section first provides background on adversarial ML and model quantization, and then presents recent related research efforts on the robustness of quantized models.

A. Background on Adversarial Machine Learning

Adversarial examples for learning-based components are generated by designing limited data modifications for the ML models using certain techniques. Essentially, a successful adversarial attack seeks two goals at once. First, the designed adversarial perturbation should be "small" enough to remain undetected by the target system. Second, the undetected attack should lead to big performance deviations in the predictor. Formally, given a trained ML model $f(\cdot)$, one well-recognized way to define an adversarial attack is the maximization of the target loss J for a given perturbation budget [13]. The budget constraint can be defined by an ϵ bounded distance $\Delta\boldsymbol{x}$ between the original data point x and the new adversarial data point $x' = x + \Delta\boldsymbol{x}$.

$$\max_{\|\Delta\boldsymbol{x}\|_p<\epsilon} J(f, x, \Delta\boldsymbol{x}) \tag{1}$$

To meet the criteria for an adversarial attack, the optimization problem in Equation 1 aims to solve for a bound-limited perturbation that can maximize the target loss function.

1) Adversarial Attack Methods: There exist two widely-used adversarial attack methods as described in the literature: *Fast Gradient Sign Method (FGSM)* and *DeepFool*.

FGSM (Fast Gradient Descent Method) [9] is one of the most widely-used adversarial attack methods to maximize a target loss function in one step. This method is intuitive and straightforward. The attacker adds fixed magnitude perturbations according to the gradient directions of the input to maximize a target loss function. The Project Gradient Descent (PGD) method improves upon FGSM by using a random starting point with smaller iterative gradient steps [14].

DeepFool [15] is an untargeted attack technique that pushes the data sample closer to the classification decision boundary. The construction of DeepFool is based on the simplified assumption that a deep neural network linearly separates different classes with a hyperplane in high dimensional space. Since neural networks are not actually linear, at each time step, it takes one step towards the target class. This search repeats and terminates only when a true adversarial example is found. As a result, this approach formulates an optimized attack method for the L_2 Norm distance metric.

2) Universal Adversarial Perturbation (UAP): The adversarial attack methods described above are all focused on individual data inputs, which makes them computationally inefficient and practically difficult for time-sensitive scenarios like cloud/edge offloading. Given the high transferability of adversarial examples, researchers have found the existence of a single small image perturbation that can successfully fool a state-of-the-art deep neural network classifier with high probability on all inputs [10].

The UAP algorithm is efficient and uses an iterative approach, which keeps pushing the minimal perturbation towards the high dimensional decision boundary between data samples using the computation method of FGSM or DeepFool mentioned above. Iterations of computations are used to aggregate the current instance of the more universal perturbation. The goal of a successful UAP is not at fooling all data points but attempting to maximize the fooling rate over the data batch. Thus, successful UAPs can push most data samples out of their original decision boundaries with high probability. UAPs exist for other applications like audio also [16]. Further, targeted UAP attacks are also proposed to mislead the predictor for a specific target class [17].

B. Model Quantization

Since many service components must execute at the edge, machine learning models must satisfy the capacity and energy constraints of edge resources. Thus, to satisfy the constraints of edge devices, ML models with compact network architectures and parameter representations are needed, which is achieved through techniques such as model compression.

There exist diverse techniques for model compression like model quantization, pruning and layer decomposition [18]. Model quantization, which we use in our research, refers to the process of reducing the number of bits to represent a value. In deep learning, the default numerical format to date has been 32-bit floating point (FP32). However, it has been demonstrated that quantization can significantly reduce bandwidth and storage usage, and improve energy efficiency on edge hardware [19]. Moreover, aggressive lower bit-width quantization options like binary models can further make use of bitwise operations to accelerate computations [20].

A quantized model can be obtained from post-training quantization (PTQ) followed by pruning the normal full-precision models [3] or via quantization-aware training (QAT) [21]. Past research has shown that weights and activations represented using 8-bit integers have a higher chance of being compressed without obvious performance loss [22]. These relatively conservative quantized models can usually be generated directly from a pre-trained full-precision model using quantization mapping functions with little performance loss [3]. More aggressive options of even lower bit-widths (≤ 4 bits) [21] or more flexible value representations [23] have also shown much progress.

C. Related Work on Model Quantization and Adversarial Robustness

We now present prior efforts at the intersection of adversarial machine learning and model compression, however, the available literature is scarce. In prior efforts, researchers have related model quantization with adversarial robustness [24], [11]. After discovering the potential advantage of quantized models, researchers have even proposed different quantization strategies as general solutions to defend against adversarial attacks [12], [25].

A recent work [26] considers the transfer of adversarial examples across neural networks with different levels of low-bit quantization models and observes that adversarial examples from lower-bit models get transferred to higher-bit or full precision models easily. For the UAP settings, researchers show preliminary efforts on evaluating them on compressed models [27]. One potential issue stems from its selected quantization method of APoT (Additive Power-of-Two) [23] using a non-uniform mapping, which is not generally supported on edge hardware devices. In contrast, we focus on the more general, hardware-aware uniform quantization setting.

In summary, we posit that the adversarial robustness issue on cloud-edge hardware requires further investigations into the following questions, which is the focus of this paper:

1) Although research on model quantization with adversarial robustness exists, these are mostly point solutions using input-dependent settings raising the question whether such attack settings are too ideal without considering hardware resource limitations?
2) Past research has focused heavily on the transferability of adversarial examples from full precision to quantized or from quantized to quantized raising the question whether more adaptive attack evaluations can be designed?
3) The potential adversarial risk on various model settings raises the question whether models can be deployed at the edge in a resilient way to mitigate potential adversarial impacts without the need to reinvent the effort for every edge-based data-driven application?

III. Adversarially Robust Data-driven Services: Design and Implementation

We now describe the design of our reusable, cloud-hosted service to realize adversarially robust data-driven services running in the cloud/edge. Our approach comprises two phases as shown in Figure 2: an offline phase and an online phase of which the offline phase is more significant and drives the deployment of the appropriate adversarially robust quantized model in the run-time infrastructure of the service.

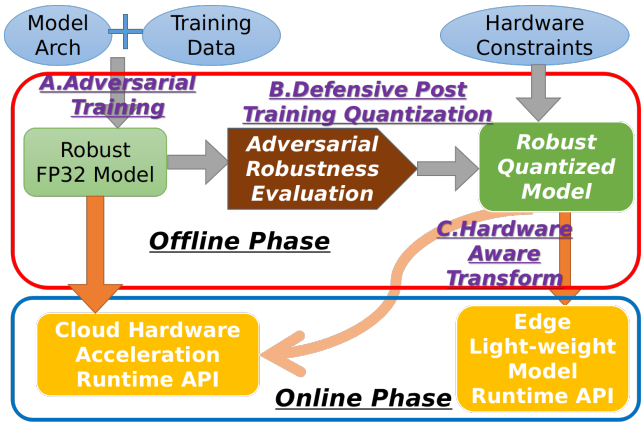

Figure 2: Proposed Workflow for deploying robust quantized models for cloud/edge computing. Full-precision models preserve performance on the cloud. Quantized models on the cloud are sometimes preferred for increasing computation throughput. With limited hardware resources, quantized versions of robust models are preferred at the edge. All compute intensive model preprocessing steps are conducted offline and only model inference is executed online and hence no extra overhead is induced at run-time. Our work validates the efficiency of state-of-art adversarial training in Step A and further proposes the method towards quantized robust models in Step B. These together bridge the gap between robust ML models and efficient hardware deployment in cloud/edge environments (Step C).

In Step A of the offline phase, a user-supplied ML model along with training data is submitted to our system to undergo adversarial training. In Step B, which is the key novel idea in our work, this adversarially trained full, robust model then undergoes quantization and robustness evaluation to form robust quantized models of different precision levels. Subsequently, in Step C, which is the online phase, depending on the constraints imposed by the underlying edge hardware, i.e., the intended deployment target, an appropriate robust, quantized model is synthesized and deployed in the run-time to perform the prediction/inference operations at the edge.

Step A can make use of robust models generated by adversarial training techniques [14], which is the mainstream defense technique against adversarial attacks. Step C involves hardware synthesis, which relies on hardware-specific toolchains from hardware vendors, such as the PyTorch machine learning framework available on NVIDIA GPUs. Our work validates the efficiency and necessity of state-of-art adversarial training in Step A and further proposes the generation procedure of quantized robust models in Step B. These together bridge the gap between robust ML models and efficient hardware deployment in cloud/edge environments.

A. Synthesizing Adversarially Robust, Hardware-aware Quantized Models

We now present our approach to realizing adversarially robust quantized models, i.e., Steps A through C of Figure 2. From Section II-A we know that an adversarial perturbation leads to large expansive increase in a target loss function with limited input modification. As a result, to make a model more robust we can make its potential loss function non-expansive under constraints. In [12], the authors control quantized weights in adversarial training to suppress the network's amplification effect across layers so as to mitigate potential adversarial impacts. Similarly, researchers from [25] proposed an iterative model compression framework combining quantization and pruning during optimization to train more robust models.

These past efforts show possible ways to get more robust quantized models. However, they are computationally expensive and cannot be combined with other defense techniques. Thus, we devise a novel approach where instead of training from scratch, we obtain a robust, quantized model by directly quantizing a full precision but adversarially robust model $\boldsymbol{f}_{rob}$ that is trained using techniques like adversarial training [28]. The consequence is that our approach is compatible with all state-of-the-art adversarial training defenses without any impediments. Note that the success of this approach is predicated on the fact that the post-training quantization of the model preserves the adversarial robustness of its original full precision model. To the best of our knowledge, the potential impacts of quantization on adversarially trained models have not been previously studied thereby highlighting the novelty of our approach.

A neural network classifier computes likelihood scores for output classes and the class with the high score becomes the final classification result. Generally speaking, robust models are non-expansive [12] in prediction likelihood scores. That is, given a bound-limited input variation, the magnitude changes in the output class scores from the robust models are also bound-limited. This in practice can be visualized through empirical experiments as shown in Figure 3. The confidence of the classification is mostly determined by prediction scores for the first and second most likely output classes. As seen from the figure, based on the same network architecture the prediction scores for the maximum and the second maximum likelihood classes are more separated for the standard model but the scores are smoothed for the robust model [29]. This further illustrates the trade-off between the natural robustness (prediction confidence maximization) and adversarial robustness (prediction confidence smoothening) [30].

Based on these insights, we illustrate the impact of quantization on robust models.There is no formal definition of what is a robust model under adversarial attacks. One important reason is the diversity of defense policy deployment phases that take into account the potential adversarial impacts. Given the evasion

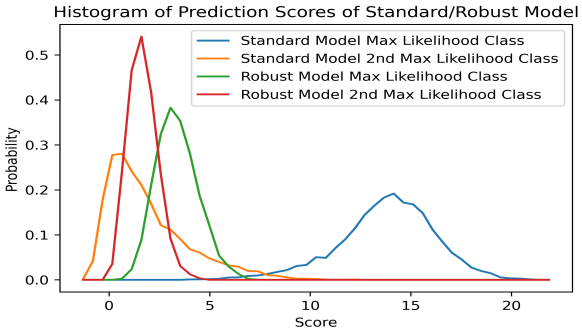

Figure 3: Comparison of prediction scores for 1st and 2nd most likely classes between a standard WideResNet model and a robust model based on the same WideResNet architecture. A neural network classifier computes likelihood scores for classes and the highest score class would be the classification output.

attack setting [31], defenses can be conducted as input data processing [32], [33], model internal robustness augmentation with adversarial training [14], [30] and more flexible model prediction procedures [34], [35]. Here, we put our focus on the adversarial training which uses the most straightforward model prediction procedure with one single prediction phase.

In this way, we are discussing the single model, single phase prediction robustness under strong enough adaptive attacks [36], [13]. Generally speaking, robust models are generated by the technique of adversarial training. Based on their design principles and actual efficiency, adversarial training has evolved for three generations. The first generation tries to insert adversarial examples into the training dataset directly [9]. This proves to be efficient for single-step FGSM attacks but vulnerable to multi-step attack methods. Later, the second generation adversarial training induces the iterative attacks for adversarial training sample generation [14]. The adversarial examples are updated in training epochs for data batches and intermediate models. These days, researchers put more attention on the empirical trade-off between adversarial robustness and natural robustness [37], [38]. This leads to the state-of-the-art regularization based adversarial training method that is seeking smoother decision boundaries for a more balanced performance between adversarial robustness and natural robustness [30].

As mentioned above, there have been many adversarial training methods developed so far [28] that are guided by diverse design ideologies. As past research works have shown, the error amplification of neural network models through their layers under adversarial attacks [32], [39], researchers are seeking defense in the opposite way. It has been shown in both theory and practice that balanced robust models should be achievable using methods that can impose tightened local Lipschitzness [40] in layers and augmenting them with generalization techniques like dropout [41]. Empirically, current robust models have shown the feature of being non-expansive [12] in prediction outputs. That is, given a bound-limited input variation, the output magnitude changes from robust models are also bound-limited. It is worth pointing out that even though this is hard to be formally proven, in practice the amplitude control effect can be visualized through empirical experiments as shown in Figure 3 and Figure 4.

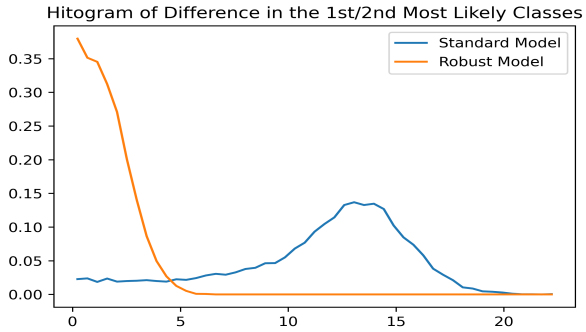

Figure 4: Prediction score differences between 1st and 2nd most likely classes of a standard WideResNet model and a robust model with the same architecture. This difference reflects the distance from the data point to its nearest decision boundary.

Here the robust model [29] is based on the same network architecture as the standard model. As seen from Figure 3, the prediction scores for the maximum and the second maximum likelihood classes are more separated for the standard model but the scores are smoothed for robust models. Figure 4 shows the prediction score differences between 1st and 2nd most likely classes on natural and robust models. This difference reflects the distance from the data point to its nearest decision boundary. This further illustrates the trade-off between the natural robustness (prediction confidence maximization) and adversarial robustness (prediction confidence smoothening) achieved by adversarial training techniques [30].

Based on these insights, we illustrate the impact of quantization on robust models. The theoretical analysis and proof is provided in the next paragraphs. As part of our approach, we present two propositions for which the theoretical proofs are provided below. The first defines the non-expansiveness of current robust neural network models. Further, based on this general characteristic, we prove that this robustness property can be preserved under quantization operations. This becomes the theoretical basis of our defense framework. Even though there has been research work that already made trials on the impact of quantization operations on robust models[42], their analysis does not show a clear definition of robust models.

Proposition 1 (Robust Models are Non-expansive)**.** *Robust models can suppress the amplification effects of deep networks by controlling the neural network's Lipschitz constant.*

For a function $f : X \to Y$, if it satisfies:

$$D_Y\left(f\left(x_1\right), f\left(x_2\right)\right) \leq \mathrm{Lip}(f) D_X\left(x_1, x_2\right), \forall x_1, x_2 \in X \tag{2}$$

for a real-valued $k \geq 0$ and some metrics D_X and D_Y , then we call f Lipschitz continuous and $\mathrm{Lip}(f)$ is defined as the Lipschitz constant of f. The Lipschitz constant describes how much the function output would change given a certain input

variation and therefore can be used as a metric for robustness under bound-limited perturbations.

Without loss of generality, a feed-forward network is composed of a series of functions:

$$f(x) = (\phi_l \circ \phi_{l-1} \circ \ldots \circ \phi_1)(x) \tag{3}$$

where ϕ_l denotes the function of one layer. In this way, the function of the ith layer has its corresponding Lipschitz constant of $\mathrm{Lip}(\phi_i)$. For the input and output of this specific layer, we have the following property:

For a neural network layer computation function $\phi_i : X_i \rightarrow Y_i$, it would satisfy:

$$D_Y(\phi_i(x_i), \phi_i(x_i + \Delta x_i)) \leq \mathrm{Lip}(\phi_i) D_X(x_i, x_i + \Delta x_i) \quad \forall x_i, x_i + \Delta x_i \in X_i \tag{4}$$

Here Δx_i denotes the perturbation caused by the adversarial input. For the very first layer, this variation equals to the input data adversarial perturbation, which means $\Delta x_i = \eta$. As a result, the overall Lipschitz constant of the whole feed-forward network is the product of its L individual layer Lipschitz constants $\mathrm{Lip}(\phi_i)$.

$$\mathrm{Lip}(f) \leq \prod_{i=1}^{L} \mathrm{Lip}(\phi_i) \tag{5}$$

We focus on the difference between clean inputs $x_1 = X$ and corresponding adversarial inputs $x_2 = X_{adv}$. We can regard a robust model as a serial combination of layer functions with suppressed Lipchitz constants so that the total amplification under bound-limited adversarial inputs can be controlled. We can imagine the most ideal case [12] when Lipchitz constants of all layers are strictly compressed ($\mathrm{Lip}(\phi_i) \leq 1.0$) so that the adversarial robustness can be fully guaranteed.

Proposition 2 (Quantization Preserves Adversarial Robustness)**.** *Given the robust neural network model formulation shown in Proposition 1, post-training quantization (PTQ) of this robust model will generate a robust quantized model with a high probability of* $\left(1 - \frac{1}{R^2}\frac{\Delta^{2L}}{12^L}\right)$*, where* $R > 1$*: a constant of prediction score ratio between two most likely classes,* Δ*: quantization step size, and* L*:number of individual layers.*

Model quantizations seek fewer representation bit-widths for model parameters θ like weights and activations. We can use a linear range mapping function as shown in Equation 6 to quantize the floating point parameters θ_f into fixed-width N_{bits} integer parameters θ_q.

$$\theta_q = \text{ round } \left\{ \left(\theta_f - min_{\theta_f}\right) \frac{2^{N_{bits}} - 1}{max_{\theta_f} - min_{\theta_f}} \right\} \tag{6}$$

As the neural network layer computations are dominated by the multiply-accumulate (MAC) operations on layer inputs, the output variation of layer i caused by the quantization step above can be bounded by a Uniform Distribution determined by the rounding step as shown:

$$Lip(\phi_i') \sim U(Lip(\phi_i) - \frac{\Delta}{2}, Lip(\phi_i) + \frac{\Delta}{2}) \tag{7}$$

Here Δ refers to the quantization step size. In linear quantization this is determined by the number of representation bits used $\Delta = \frac{1}{2^{N_{bits}-1}}$. We denote the original Lipschitz constant for the $i'th$ layer in the network before quantization as $\mu_i = Lip(\phi_i)$.

To estimate the function amplification on prediction output class change, we investigate the Lipschitz constant change of prediction scores between two most probable classes R_i for the single $i'th$ layer in the network before ($Lip(\phi_i)$) and after ($Lip(\phi_i')$) quantization using the Chebyshev's inequality:

$$\Pr(|Lip(\phi_i') - \mu_i| \geq R_i) \leq \frac{\sigma_i^2}{R_i^2} \tag{8}$$

Here σ_i^2 refers to the quantization error of layer parameters. In probability, Chebyshev's inequality guarantees that for a given probability distribution only a certain fraction of values can deviate more than a distance from the mean.

From the quantization equation Equation 6, we can see that quantizations incur an inevitable information loss with the value rounding operation. For this linear quantization setting, we can compute the quantization error power as:

$$\sigma^2_{\text{QNoise}} = E(e^2) = \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} \frac{1}{\Delta} e^2 de = \frac{\Delta^2}{12} \tag{9}$$

Moreover, this error variance should be the same for layers with same quantization settings so we would have:

$$\sigma_i^2 = \sigma^2_{\text{QNoise}} = \frac{\Delta^2}{12} \tag{10}$$

To make the current class prediction score to become invalid (no longer maximum), the perturbation needs to at least deviate the score by the ratio of R so this can be distributed to each layer as $R_i = \sqrt[L]{R}$. Here, the required R is determined by the gap between prediction scores of the maximum and $2nd$ maximum likelihood class but we can guarantee $R > 1$. Further, we get the overall perturbed probability as:

$$Pr[|Lip(f) - Lip(f')| \geq R] \leq P_i^L = \frac{1}{R^2}\frac{\Delta^{2L}}{12^L} \tag{11}$$

Consider the value range of quantization step Δ (for example $\Delta = \frac{1}{2^8-1} = \frac{1}{255}$ for the 8-bit quantized representation), even consider the extreme case of only one layer $L = 1$ and very close prediction scores for the two classes $R \approx 1$, the flipping probability computed from the above equation would be $\frac{1}{255} * \frac{1}{255} * \frac{1}{12}$. Obviously, the possibility of classification change after quantization is strictly compressed if the original model is robust under adversarial attacks.

Thus we have proved that quantization preserves the non-expansiveness of robust models by preserving the overall Lipschitz constant of the network. The flexible applications of robust models could be a potential technical path towards more robust quantized models for cloud/edge applications. This analysis might also indicate the impact of quantization

operations of models in the reverse way. It indicates simply using quantization on non-robust machine learning models would not help improve the adversarial robustness.

Compared to past methods that aim to get a robust, quantized model from scratch and extra training, we propose a simpler and novel approach based on the proposition, which can be implemented in a flexible way as shown below. This analysis drives our approach, which can be formalized as below:

$$\begin{aligned} &\min_{\boldsymbol{\theta}} \max_{x'} J\big(\boldsymbol{f}_{rob}(\boldsymbol{\theta}), x, \Delta\boldsymbol{x})\big) \\ &\text{subject to:} x' = x + \Delta\boldsymbol{x} \\ &\|\Delta\boldsymbol{x}\|_p < \epsilon \\ &W_b, A_b \in \boldsymbol{\theta} \end{aligned} \tag{12}$$

In other words, instead of training a robust, quantized model from scratch, we quantize an already adversarially robust full precision model $\boldsymbol{f}_{rob}$ obtained using adversarial training [28]. We consider this relatively simple problem to solve for an optimal quantization setting of weight bitwidth W_b and activation bitwidth A_b to maintain the robustness. Such an approach greatly reduces the search space and computational burden while maintaining model robustness.

Algorithm 1 illustrates the approach, which uses a simple search process to iteratively test potential settings for model weights and activations. We randomly select a batch of data (or full data) for evaluation (line 5) and compare the natural (line 6) and adversarial performance (lines 7-8) for a selected quantization setting (lines 3-4) on this set of data. The selection is made to maximize the adversarial robustness (classification accuracy in this case) while maintaining the natural robustness at an acceptable level (lines 9-11). Once the appropriate model is synthesized, it is transformed into a form suitable for the edge hardware using existing tools (Step C in Fig 2).

Algorithm 1 Searching for the Robustness-Driven Optimal Model Quantization Setting

Require: $\boldsymbol{X}$: data points ; $\boldsymbol{f}_{rob}$: full precision robust predictor; $J(func, x, \epsilon)$: cost function of model $func$ according to input data x; ϵ: maximum bound distance allowed to be modified for each feature (like L_∞ constraint); $\boldsymbol{quant}$: post-training quantization function, $QuantList$: quantization setting list for weight bitwidth W_b and activation bitwidth A_b; $eval(func, x)$:evaluation metric for model $func$ according to input data x.

1: $best \leftarrow 0$,$optBit \leftarrow empty$
2: **while** QuantList **do**
3: $W_b, A_b \leftarrow pop(QuantList)$
4: $\boldsymbol{f}_{robQT} \leftarrow \boldsymbol{quant}(\boldsymbol{f}_{rob}, W_b, A_b)$
5: $\boldsymbol{x} \leftarrow randomBatch(\boldsymbol{X})$
6: $natural \leftarrow eval(\boldsymbol{f}_{robQT}, \boldsymbol{x})$
7: $\boldsymbol{x}_{adv} \leftarrow argmax \nabla J(\boldsymbol{f}_{robQT}, \boldsymbol{x}, \epsilon)$
8: $current \leftarrow eval(\boldsymbol{f}_{robQT}, \boldsymbol{x}_{adv})$
9: **if** $natural > threshold$ **and** $current > best$ **then**
10: $best \leftarrow current$
11: $optBit \leftarrow [W_b, A_b]$
12: **end if**
13: **end while**
14: **return** $optBit$

B. Generating Universal Adversarial Perturbations across Quantized Models for Evaluating Robustness

Although synthesizing a configurable, robust and quantized model suitable for the underlying edge hardware is a necessary step, we also need a way to automate validation of its robustness and efficiency. In other words, we seek a broadly applicable automated attack generation procedure that can work across different precision levels that our service can support.

A recent effort in universal perturbations has focused on *attack generation* using a single model [17] and then attempts to transfer the vulnerability to other models [43]. We extend this classical UAP generation algorithm to support exploration on multiple models as shown in *Algorithm 2*. Algorithm 2 is implemented based on a white-box setting, which assumes that the attacker has full knowledge of the ML model like its architecture information and weights. This is not unrealistic as numerous ML models are open to public.

Moreover, there exist two settings based on the attack target: the *direct attacks* generate the adversarial perturbation from the testing data directly; the *indirect attacks* generate the adversarial perturbation from the training data and then add the perturbation to the testing data. To make these attacks more realistic, we mostly consider the latter setting where the attack is implemented on the training data while the generated UAPs are evaluated on the separate testing data.

Algorithm 2 Universal Perturbation using Model Ensemble

Require: $\boldsymbol{X}$: data points ; $\boldsymbol{F}$: predictor set containing predictors (with different quantization settings); $J(func, x, \Delta\boldsymbol{x})$: cost function of model $func$ according to input data x with a step size of α; $clip$: clips inputs in a range with a lower and an upper bound; α: modification step size for each iteration; ϵ: maximum bound distance allowed to be modified for each feature(L_∞ constraint).

1: $\Delta\boldsymbol{x} \leftarrow \mathbf{0}$, $i \leftarrow 0$
2: **while** $i < NumIter$ **do**
3: $\boldsymbol{f} \leftarrow randomSelect(\boldsymbol{F})$
4: $\boldsymbol{x} \leftarrow randomSelect(\boldsymbol{X})$
5: $\Delta\boldsymbol{x} \leftarrow argmax \nabla J(\boldsymbol{f}, \boldsymbol{x}, \Delta\boldsymbol{x})$
6: $\Delta\boldsymbol{x} \leftarrow clip\{\Delta\boldsymbol{x}, \Delta\boldsymbol{x} - \alpha, \Delta\boldsymbol{x} + \alpha\}$
7: $i \leftarrow i + 1$
8: **end while**
9: $\Delta\boldsymbol{x} \leftarrow clip\{\Delta\boldsymbol{x}, \Delta\boldsymbol{x} - \epsilon, \Delta\boldsymbol{x} + \epsilon\}$
10: **return** $\Delta\boldsymbol{x}$

The algorithm uses an iterative procedure (line 2). To compensate for the single precision normal UAP generation method [10], a prediction model is randomly selected (line 3) per iteration and the perturbation is gradually generated across the data points (line 4) using a small attack step (lines $5-6$). The final step projects the generated data back into its valid value range (line 9). This ensembles multiple predictors for

UAP generation and formulates an equivalent mixed-precision attack setting that is suitable for evaluation [44].

C. Realizing a Cloud-hosted UAP Guard Service

We offer our capabilities as a cloud-native service via a Kubernetes-managed composition of containerized components, each implementing an individual functionality from Figure 2. Developers of edge-based data-driven services can use our proposed procedure as follows.

The user supplies a prediction task, which is analyzed by a task analysis service to identify the exact requirements for the target model. If it belongs to the mature prediction tasks like ImageNet or CIFAR10/100, a pretrained robust model can be fetched directly. Otherwise, the service would fetch a model from a similar task and use some data that the user provides to perform adversarial transfer learning. In this way, we obtain robust full-precision models for the given prediction tasks.

Next, for the target hardware platform specified by the user, potential model quantization settings would be sent to the quantization service. One quantized model would be generated from each setting and the UAP adversarial robustness evaluation service would follow. Of course, for all model settings, their quantization and evaluation procedures can be conducted in parallel. We choose the best setting after evaluation on all potential settings. Finally, we conduct model transformations for target hardware platforms and let the user deploy them. The main computation burden lies in the offline robust model processing and evaluation part. For widely used datasets like ImageNet/CIFAR10/100, pre-trained robust models can be found straightforwardly so no time consumption. For robust model evaluation step, the time consumption is proportional to potential quantization options. There is no additional time cost for the final robust quantized model deployment phase.

IV. Research Evaluation

This section evaluates our framework for the claims we make. Our evaluations first illustrate the vulnerability of normal quantized neural networks under UAPs followed by the efficiency of obtaining robust quantized neural networks using post-training quantization methods. We then show the feasibility of deploying robust quantified models at the edge.

A. Framework Implementation and Experimental Setup

For the post-training quantization (***PTQ***) approach used by our framework, we use the BrainChip Akida CNN2SNN Toolkit [45], [2] which is a light-weight model quantization toolkit that is compatible with the Keras ML framework [46].

We consider three widely used computer vision datasets of CIFAR10/CIFAR100/SVHN. CIFAR10 and CIFAR100 datasets [47] are 32x32 colour images in 10/100 object classes, with 6000/600 images per class. It comprises 50,000 training images and 10,000 test images. SVHN [48] is a benchmark dataset containing over 600,000 labeled digits cropped from Street View images. The images included in these datasets are RGB images with 8-bit pixel values ranging from $0-255$. Perturbation magnitudes can be added in this range and a change of $8/255$ leads to approximately 3.1% perturbations.

We incorporate different ML model architectures that prove to be well-performed in full-precision mode as follows:

- Akida DS-CNN [45] is a MobileNet [49] architecture model with fewer layers. It utilizes depthwise separable convolutions to reduce the model size and complexity. This model is used for CIFAR10 attack evaluation.
- ResNet [50] induces skip connections allowing efficient training of very deep network models. This model is used as the natural model for the SVHN dataset.
- WideResNet [51] shows state-of-the-art performance on many object classification datasets and from which many robust models are also adversarially trained [28]. This architecture is used for all three datasets, i.e., CIFAR10/CIFAR100/SVHN.

B. Results from UAP Adversarial Attacks

Here we answer the question raised in Section I for which we conduct a detailed investigation of adversarial impacts on the CIFAR10 dataset. By using our universal attack procedure (see Algorithm 2), we show the existence of universal adversarial perturbations across different quantization precision levels. Examples of these kinds of perturbations are shown in Figure 5. The attack strengths (L_{inf}) increase from left to right: $original$, 1.6%,3.1%, 4.7%,6.3%, 7.8%,10.2%, 12.5%. We observe that the perturbations become relatively obvious when the attack strength is higher than 6% level but even on the strongest 10% level we can only observe some texture noise but the original structural information of the image is still well-preserved.

The post-training quantization (PTQ) model prediction results under white-box settings are shown in Figures 6 and 7. We analyze these results under different predictor and attack settings. To ease the notation, we use the abbreviation "wXaY" to indicate the quantization setting with X bits of *weight* and Y bits of *activation.* For comparison purposes, the accuracy of the "ensemble" is defined as the mean of the classification accuracy of multiple models for white-box experiments.

Under white-box settings, the vulnerability expands for both direct test set or indirect training set attacks under increasing attack strengths. Quantized models of w2a2, w3a2 and w3a3 show higher accuracy on a wide range of adversarial perturbations. The results also indicate that for the full-precision (fp32) model, the universal perturbation from the training set data seems to be more aggressive than the test set data.

The results also reveal that model quantization does not provide any predictable improvement in adversarial robustness. The diversity in the results stemming from different settings shows the significant need for designing a systematic evaluation workflow that we have automated in this research.

C. Evaluating Generated Robust, Quantized Models

We illustrate the efficiency of the proposed defense settings on more datasets. We show the impacts of perturbations

[2]We thank Brainchip Inc for granting us academic use of codes and models from the Akida toolkit. Code:https://github.com/dustinjoe/Universal-Adversarial-Perturbation-on-Quantized-Models

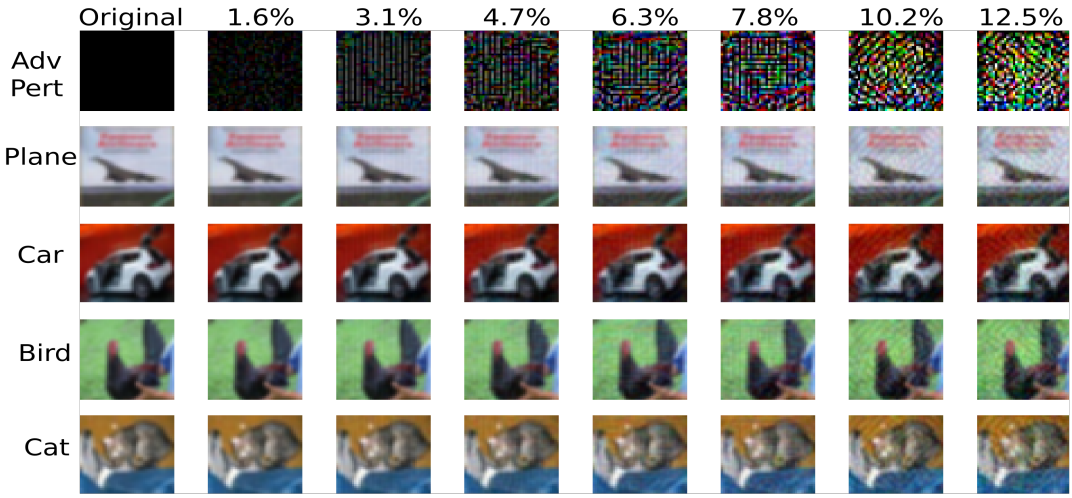

Figure 5: Example Universal Perturbations on CIFAR10 Classes: airplanes, cars, birds, cats. Attack strengths (L_{inf}) increase from left to right: $original, 1.6\%, 3.1\%, 4.7\%, 6.3\%, 7.8\%, 10.2\%, 12.5\%$. We can observe some texture noise from images with the strongest perturbations from comparisons with raw images. But they still preserve original data structural information and are easy to recognize by human eyes.

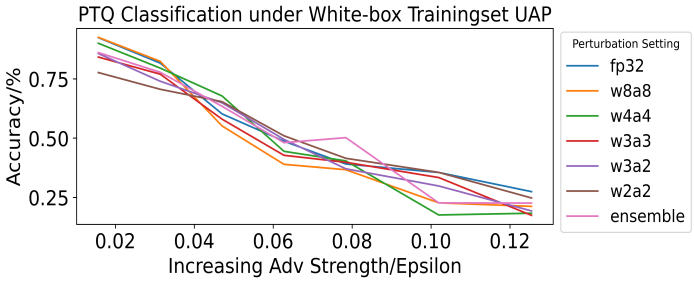

Figure 6: Accuracy of PTQ Model under White-box Training Data Attack

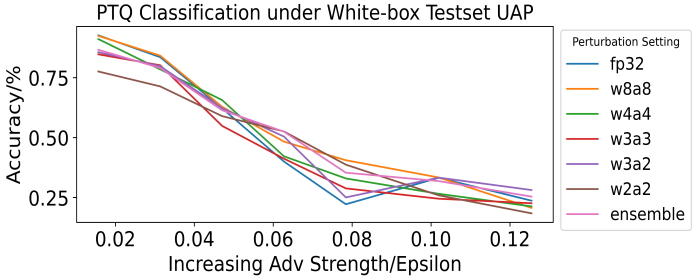

Figure 7: Accuracy of PTQ Model under White-box Testing Data Attack

generated from white-box settings. We then show the efficiency of defending against such kinds of adversarial perturbations using post-training quantization of robust models. It is worth pointing out that we focus on the defense in the training set adversarial settings because these attacks are shown to be stronger and more practical in realistic settings.

Fortunately, even though the state-of-the-art defensive adversarial training methods [14] are based mostly on iterative white-box attack settings, the robustness of these models are highly preserved under universal adversarial perturbations. We obtain adversarially trained robust full-precision models with state-of-the-art performances [28] for the proposed defensive quantization evaluation as follows:

- *CIFAR10:* Pre-trained robust model from research work [52] induces more adaptive regularizations on the decision boundary of the classifier.
- *CIFAR100:* Pre-trained robust model from research work [53] demonstrates how adversarial robustness can benefit from combining larger models, Swish/SiLU activations and model weight averaging.
- *SVHN:* Transfer learning from a pre-trained robust CIFAR10 model from research work [54] demonstrates how adversarial robustness can benefit from semi-supervised learning with some extra data.

Overall, we use diverse model architectures for different tasks in our experiments. The natural model of CIFAR10 is a standard WideResnet network without adversarial training. The natural model of CIFAR100 is an adversarially-trained model on a different norm distance metric, which shows the incompatibility of robust models from different norm metrics. The robust model of SVHN is based on the transfer learning of a robust model of CIFAR10. We test our proposed defensive quantization approach using these robust models on corresponding datasets. Robustness validation results of these three models are shown in Figure 8, Figure 9 and Figure 10, respectively.

We observe that the linear quantization of weight and activation values preserve model robustness. Moreover, for these robust image classification models, the optimal quantized representations vary across different datasets. On CIFAR10 and CIFAR100, the quantized representation using $optBit = [W_b \leftarrow 7, A_b \leftarrow 7]$ bits in weights and activation shows the best adversarial robustness with the acceptable natural robustness. On SVHN, the quantized representation using

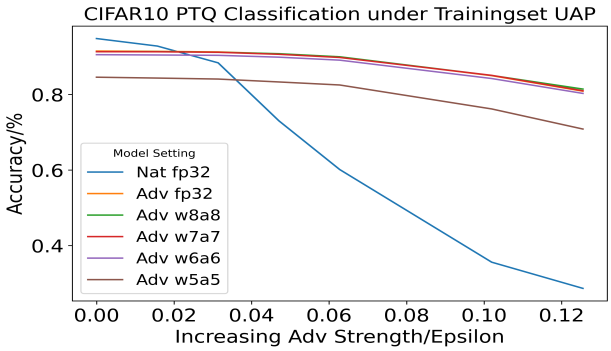

Figure 8: Defensive Post-training Quantization on CIFAR10.

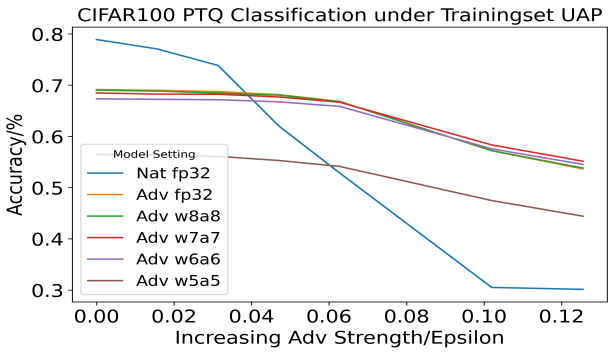

Figure 9: Defensive Post-training Quantization on CIFAR100.

$optBit = [W_b \leftarrow 6, A_b \leftarrow 6]$ bits in weights and activation shows the best adversarial robustness with a good natural robustness. Relative high bitwidth quantizations (≥ 6 bits) show very similar model inference performances under increasing adversarial strengths. For these models, it is hard to differentiate on these figures. Optimal quantization settings are selected by comparing precise accuracy values. These results show the efficiency and scalability of our adversarial training with post-training quantization approach used in the defensive model deployment framework for cloud/edge environments.

D. Validation on Prototypical Edge Hardware Deployment

To understand and validate the practicality of deploying robust models on the cloud and edge, we provide a prototypical

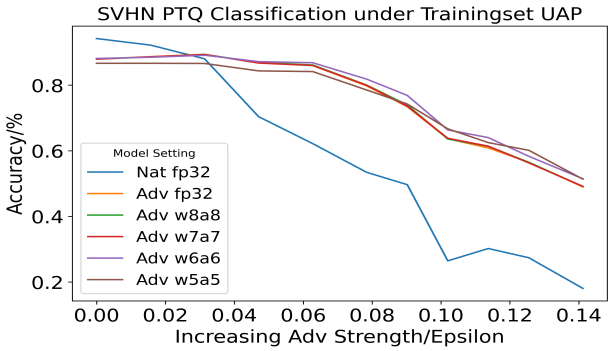

Figure 10: Defensive Post-training Quantization on SVHN.

hardware deployment of these models on a server GPU and an edge platform as follows:

- **Cloud Hardware:** The cloud inference deployment was conducted on a mobile workstation (server) with a 4 Core/8 Thread Intel I7-7700HQ processor, 64GB RAM and a 2560-core NVIDIA P5000 Mobile Pascal GPU with 16GB GDDR5X VRAM.
- **Edge Hardware:** The edge inference deployment was conducted on an NVIDIA Jetson Nano Development Board with an ARM A57 Quad-core CPU, 4GB RAM and an embedded 128-core Maxwell GPU.

Even though hardware configurations for cloud and edge deployment are relatively different, the hardware accelerations are all based on NVIDIA GPUs. In this way, some cross-platform ML engines like PyTorch can be reused. We conduct model deployment profiling on a batch inference of 32 data inputs and obtain the average value of 100 executions when measuring the memory usage and the latency. Experimental results are shown in Table I.

There still exist some key differences in the prototypical deployment workflows. On the cloud side, we only consider 32-bit full-precision (FP32) models to guarantee the completeness of model performance. Here, the quantized robust model settings shown on the cloud side are based on the 'fake' quantization simulation just as quantization-aware training does. In this way, model parameters are still stored as 32-bit floating point values that are rounded to the corresponding quantization value range. That is the reason why the model size of standard 'WidRes28-10' and quantized robust models are the same on the cloud side. Since we are deploying the same model on the cloud and edge, they share the same number of model parameters leading to the same level of computations (MACs).

It is worth pointing out that computations with quantization settings are strictly limited by the underlying hardware support. For the cloud side P5000 Pascal GPU, it has internal precision computation support of FP32, FP16 and as low as INT8 (8-bit integer). But INT8 computations are based on the TensorRT optimization framework [55] developed by NVIDIA and are not supported by all devices. For the edge side Jetson Nano, it only has FP32 and FP16 support given its hardware architecture. Thus, better quantization support has become a demanding trend for new edge devices. Newer devices like Google Coral Stick or Board have incorporated INT8 computations. Computation devices like FPGAs (field programmble gate array) have the most flexible bit-width support but they also need deep knowledge and expertise in FPGA deployment.

For the relatively flexible quantization settings like 7-bit or 6-bit that we consider, we store models in higher precisions (FP32, FP16, INT8) and then round these parameter values to lower bitwidth range for equivalent computations. Even though Jetson Nano devices do not have INT8 hardware support, their half-float FP16 representions are shown to be efficient. According to the results in Table I, the inference latency of FP16 model already reaches the inference latency level of the full precision model from the cloud-side. This means that the deployment of robust models either on cloud or edge would

Table I: Prototypical Hardware Deployment

Platform	Model Setting	Compute Core	Model Format	Precision	Size /mb	#Param	#MACs	Memory Usage/mb	Latency /ms
Cloud GPU	WideRes28-10	GPU	Pytorch	FP32	292.1	38.12M	257.26G	396	1.4
	Rob1(w8a8,w7a7)	GPU	Pytorch	FP32	292.1			436	1.42
Jetson Nano	WideRes28-10	GPU	Pytorch	FP32	292.1			686	12.7
	Rob1_w7a7_FP16	GPU	Pytorch	FP16	219.2			186	1.19

not incur extra cost. We also measure the memory usage of the model inference, where on the cloud side this memory usage is measured by the peak VRAM usage in the GPU. On the edge side, Jetson Nano has a unified shared memory for CPU and GPU so the memory usage on the edge belongs to both RAM and VRAM [4]. This explains why the same 32-bit full-precision model requires more memory usage on the edge side. There are many metrics for machine learning deployment efficiency evaluation [18], we only choose model memory size usage and adversarial accuracy in our research here.

To summarize, we show the potential for deploying robust models in both the cloud and edge deployments. We choose popular commercial hardware devices to show a prototype of practical model deployment workflow. In contrast, the lack of good hardware support also limits the capability of deploying these models. Moreover, it can be anticipated that the high demand for reliable model deployment on the edge side will lead to a large market in more efficient edge hardware accelerators and more user-friendly software tools [4].

V. Conclusions

Recap: This paper presented a reusable and configurable, cloud-native service that transforms a user-supplied deep learning machine learning (ML) model into an adversarially robust quantized model suitable for the specified edge hardware. The paper shows that there is no general guarantee of adversarial robustness improvement from pure model quantization operations. Experimental evaluations reveal that traditional quantized models are vulnerable to universal perturbations and moreover, the perturbations from some quantized models are even more transferable. The work highlights the need for combining state-of-art adversarial training with model quantization techniques. Further, this work is the first to propose the integrated generation procedure of quantized robust models combining model evaluation and quantization using adversarially trained models. Such adversarially trained quantized neural networks continue to be compact and efficient making them useful in realizing edge-based services.

Limitations and Future Work: Our future work will address current limitations in the presented work as follows: (1) The post-training method we explored cannot generate models for aggressive quantization settings like binary neural networks. Our future work will therefore explore more systematic quantization-aware adversarial training for more flexible model settings [56]; (2) The deployable quantized models in the current work are generated offline and hence cannot be adapted dynamically as operating conditions may change. Accordingly, we will investigate dynamic adaptations; (3) Distributed model training [57] and inference [58] are increasingly prevalent in edge scenarios and hence there is a need for distributed, interacting quantized models with potentially different precision levels; (4) We currently put attention on adversarial threats on object classifiers. Other application settings like object detection [59] would also under potential adversarial threats; and (5) Investigating the use of machine learning compilation and optimization tools like TVM [6] for generating more flexible robust quantized models remains an open problem.

Acknowledgment

Work supported in part by The National Science Foundation's Smart and Connected Communities Program under Award 1952029. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of these sponsors.

References

[1] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[2] M. Chiang and T. Zhang, "Fog and iot: An overview of research opportunities," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, 2016.

[3] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[4] X. Zhou, R. Canady, S. Bao, and A. Gokhale, "Cost-effective hardware accelerator recommendation for edge computing," in *3rd {USENIX} Workshop on Hot Topics in Edge Computing (HotEdge 20)*, 2020.

[5] K. Wang, Z. Liu, Y. Lin, J. Lin, and S. Han, "Haq: Hardware-aware automated quantization with mixed precision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 8612–8620.

[6] T. Chen, T. Moreau, Z. Jiang, L. Zheng, E. Yan, H. Shen, M. Cowan, L. Wang, Y. Hu, L. Ceze *et al.*, "{TVM}: An automated end-to-end optimizing compiler for deep learning," in *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*, 2018, pp. 578–594.

[7] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, "Once-for-all: Train one network and specialize it for efficient deployment," *arXiv preprint arXiv:1908.09791*, 2019.

[8] Z. Xu, H. S. Shah, and U. Ramachandran, "Coral-pie: A geo-distributed edge-compute solution for space-time vehicle tracking," in *Proceedings of the 21st International Middleware Conference*, 2020, pp. 400–414.

[9] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples (2014)," *arXiv preprint arXiv:1412.6572*, 2014.

[10] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," *arXiv preprint*, 2017.

[11] K. Duncan, E. Komendantskaya, R. Stewart, and M. Lones, "Relative robustness of quantized neural networks against adversarial attacks," in *2020 IEEE International Joint Conference on Neural Networks*. IEEE, 2020.

[12] J. Lin, C. Gan, and S. Han, "Defensive quantization: When efficiency meets robustness," *arXiv preprint arXiv:1904.08444*, 2019.

[13] N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, and A. Kurakin, "On evaluating adversarial robustness," *arXiv preprint arXiv:1902.06705*, 2019.

[14] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[15] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2574–2582.

[16] S. Abdoli, L. G. Hafemann, J. Rony, I. B. Ayed, P. Cardinal, and A. L. Koerich, "Universal adversarial audio perturbations," *arXiv preprint arXiv:1908.03173*, 2019.

[17] H. Hirano and K. Takemoto, "Simple iterative method for generating targeted universal adversarial perturbations," *Algorithms*, vol. 13, no. 11, p. 268, 2020.

[18] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[19] W. Dally, "High-performance hardware for machine learning," *NIPS Tutorial*, vol. 2, 2015.

[20] Xilinx Incorporation, "Quantized Neural Networks (QNNs) on PYNQ ," {https://github.com/Xilinx/BNN-PYNQ/}, 2017, [Online; accessed 19-July-2019].

[21] A. Pappalardo, "Brevitas pytorch library for quantization-aware training," https://github.com/Xilinx/brevitas, Xilinx Research Labs, 2020, accessed September 20, 2020.

[22] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[23] Y. Li, X. Dong, and W. Wang, "Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks," *arXiv preprint arXiv:1909.13144*, 2019.

[24] A. Galloway, G. W. Taylor, and M. Moussa, "Attacking binarized neural networks," *arXiv preprint arXiv:1711.00449*, 2017.

[25] S. Gui, H. N. Wang, H. Yang, C. Yu, Z. Wang, and J. Liu, "Model compression with adversarial robustness: A unified optimization framework," in *Advances in Neural Information Processing Systems*, 2019, pp. 1285–1296.

[26] R. Bernhard, P.-A. Moellic, and J.-M. Dutertre, "Impact of low-bitwidth quantization on the adversarial robustness for embedded neural networks," in *2019 International Conference on Cyberworlds (CW)*. IEEE, 2019, pp. 308–315.

[27] A. G. Matachana, K. T. Co, L. Muñoz-González, D. Martinez, and E. C. Lupu, "Robustness and transferability of universal attacks on compressed models," *arXiv preprint arXiv:2012.06024*, 2020.

[28] F. Croce, M. Andriushchenko, V. Sehwag, N. Flammarion, M. Chiang, P. Mittal, and M. Hein, "Robustbench: a standardized adversarial robustness benchmark," *arXiv preprint arXiv:2010.09670*, 2020.

[29] D. Wu, S.-T. Xia, and Y. Wang, "Adversarial weight perturbation helps robust generalization," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[30] H. Zhang, Y. Yu, J. Jiao, E. P. Xing, L. E. Ghaoui, and M. I. Jordan, "Theoretically principled trade-off between robustness and accuracy," *arXiv preprint arXiv:1901.08573*, 2019.

[31] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," *Pattern Recognition*, vol. 84, pp. 317–331, 2018.

[32] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and J. Zhu, "Defense against adversarial attacks using high-level representation guided denoiser," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1778–1787.

[33] C. Guo, M. Rana, M. Cisse, and L. Van Der Maaten, "Countering adversarial images using input transformations," *arXiv preprint arXiv:1711.00117*, 2017.

[34] X. Liu, M. Cheng, H. Zhang, and C.-J. Hsieh, "Towards robust neural networks via random self-ensemble," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 369–385.

[35] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," *arXiv preprint arXiv:1711.01991*, 2017.

[36] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *International conference on machine learning*. PMLR, 2020, pp. 2206–2216.

[37] A. Fawzi, H. Fawzi, and O. Fawzi, "Adversarial vulnerability for any classifier," *Advances in neural information processing systems*, vol. 31, 2018.

[38] A. Raghunathan, S. M. Xie, F. Yang, J. Duchi, and P. Liang, "Understanding and mitigating the tradeoff between robustness and accuracy," *arXiv preprint arXiv:2002.10716*, 2020.

[39] P. Panda, "Quanos: adversarial noise sensitivity driven hybrid quantization of neural networks," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, 2020, pp. 187–192.

[40] Y. Huang, H. Zhang, Y. Shi, J. Z. Kolter, and A. Anandkumar, "Training certifiably robust neural networks with efficient local lipschitz bounds," *Advances in Neural Information Processing Systems*, vol. 34, pp. 22 745–22 757, 2021.

[41] Y.-Y. Yang, C. Rashtchian, H. Zhang, R. R. Salakhutdinov, and K. Chaudhuri, "A closer look at accuracy vs. robustness," *Advances in neural information processing systems*, vol. 33, pp. 8588–8601, 2020.

[42] C. Song, R. Ranjan, and H. Li, "A layer-wise adversarial-aware quantization optimization for improving robustness," *arXiv preprint arXiv:2110.12308*, 2021.

[43] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," *arXiv preprint arXiv:1611.02770*, 2016.

[44] O. Sagi and L. Rokach, "Ensemble Learning: A Survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1249, 2018.

[45] BrainChip Incorporation, "The akida neural processor cnn2snn toolkit," 2020. [Online]. Available: https://doc.brainchipinc.com/user_guide/cnn2snn.html

[46] Keras Team, "Train a simple deep cnn on the cifar10 small images dataset." 2019, [Online; accessed 19-Mar-2020]. [Online]. Available: https://keras.io/examples/cifar10_cnn/

[47] A. Krizhevsky, V. Nair, and G. Hinton, "The cifar-10 dataset," *online: http://www. cs. toronto. edu/kriz/cifar. html*, vol. 55, 2014.

[48] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," 2011.

[49] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[50] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[51] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[52] R. Rade and S.-M. Moosavi-Dezfooli, "Helper-based adversarial training: Reducing excessive margin to achieve a better accuracy vs. robustness trade-off," in *ICML 2021 Workshop on Adversarial Machine Learning*, 2021.

[53] S. Gowal, C. Qin, J. Uesato, T. Mann, and P. Kohli, "Uncovering the limits of adversarial training against norm-bounded adversarial examples," *arXiv preprint arXiv:2010.03593*, 2020.

[54] Y. Carmon, A. Raghunathan, L. Schmidt, P. Liang, and J. C. Duchi, "Unlabeled data improves adversarial robustness," *arXiv preprint arXiv:1905.13736*, 2019.

[55] NVIDIA Incorporation, "8-bit inference with tensorrt," 2017. [Online]. Available: https://on-demand.gputechconf.com/gtc/2017/presentation/s7310-8-bit-inference-with-tensorrt.pdf

[56] Y. Guo, T. Ji, Q. Wang, L. Yu, and P. Li, "Quantized adversarial training: An iterative quantized local search approach," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 1066–1071.

[57] S. Zhou, B. A. Landman, Y. Huo, and A. Gokhale, "Communication-efficient federated learning for multi-institutional medical image classification," in *Medical Imaging 2022: Imaging Informatics for Healthcare, Research, and Applications*, vol. 12037. SPIE, 2022, pp. 6–12.

[58] T. Mohammed, C. Joe-Wong, R. Babbar, and M. Di Francesco, "Distributed inference acceleration with adaptive dnn partitioning and offloading," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 854–863.

[59] R. Canady, X. Zhou, Y. Barve, D. Balasubramanian, and A. Gokhale, "Adversarially robust edge-based object detection for assuredly autonomous systems," in *2022 IEEE International Conference on Assured Autonomy (ICAA)*. IEEE, 2022, pp. 97–106.